**Computer assignment 2**

**Data mining and networks**

1 Balance scale classification

We can see from the data description that this data set to research psychological experiment has four attributes: left weight, left distances, right weight, right distance. These data can be classified into three classes: B ( 49 balanced ), L (288 left), R(288 right). In total there are 625 data.

Since the class name is B, L, R which is not suitable for matlab to deal with. Try to use number to represents three classes. That is B: 2, L: 12 and R: 18.

1. Decision tree

A decision tree is a classifier expressed as a recursive partition of the instance space. The decision tree consists of nodes that form a rooted tree, meaning it is a directed tree with a node called “root” that has no incoming edges. A node with outgoing edges is called a test node. All other nodes are called leaves. In a decision tree, each internal node splits the instance space into two or more subspaces according to a certain discrete function of the input attributes values.

Here we use relative information gain for selection of the most informative attribute.

Formula:

In Balance scale classification, we assume that the class B,L, and R is Y. The four attributes are X. From the origin data, all value of every attribute is real number, from 1 to 5.

At first, select randomly 20% for the test set, so there are 80% for learning set which is used for constructing decision tree.(nearly 500)

Secondly, use RIG to select most informative attribute. In order to calculate RIG, we should know the probability distribution of Y. Since Y only has three possibility B,L, and R, so we can calculate the entropy of Y. Then, calculate the average specific conditional entropy of Y, H(Y|X).

where H(Y|X=v)=the entropy of Y among only those records in which X has value v.

Next, choosing the attribute which has the largest relative information gain is a node to split. According to the matlab result:

Attribute 1: left weight; attribute value:1,2,3,4,5, Class: B( 2), L(12) and R(18)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Left weight  value | 1 | 2 | 3 | 4 | 5 | RIG |
| Class 2 | 0.08 | 0.088 | 0.07 | 0.08 | 0 | 0.0845 |
| Class 12 | 0.136 | 0.344 | 0.5040 | 0.616 | 0 |
| Class 18 | 0.784 | 0.568 | 0.424 | 0.304 | 0 |

Attribute 2: left distance; attribute value:1,2,3,4,5, Class: B( 2), L(12) and R(18)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Left weight  value | 1 | 2 | 3 | 4 | 5 | RIG |
| Class 2 | 0.08 | 0.09 | 0.07 | 0.08 | 0.08 | 0.1025 |
| Class 12 | 0.09 | 0.28 | 0.44 | 0.55 | 0.64 |
| Class 18 | 0.83 | 0.63 | 0.49 | 0.37 | 0.28 |

Attribute 3: right weight; attribute value:1,2,3,4,5, Class: B( 2), L(12) and R(18)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Left weight  value | 1 | 2 | 3 | 4 | 5 | RIG |
| Class 2 | 0.09 | 0.1 | 0.08 | 0.09 | 0.04 | 0.1146 |
| Class 12 | 0.74 | 0.5 | 0.35 | 0.23 | 0.18 |
| Class 18 | 0.17 | 0.4 | 0.57 | 0.68 | 0.78 |

Attribute 4: right distance; attribute value:1,2,3,4,5, Class: B( 2), L(12) and R(18)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Left weight  value | 1 | 2 | 3 | 4 | 5 | RIG |
| Class 2 | 0.09 | 0.1 | 0.08 | 0.09 | 0.04 | 0.1146 |
| Class 12 | 0.74 | 0.5 | 0.35 | 0.23 | 0.18 |
| Class 18 | 0.17 | 0.4 | 0.57 | 0.68 | 0.78 |

It is clear to see that attribute 1 and 2 give us less information. And attribute 3,4 have the same RIG. I choose attribute 3 right weight as a node to implement next split.

The first step decision tree is as follows:

Class: B, L, R

Root

0.08(B),0.4(L),0.52(R)

RW=5

0.04, 0.18, 0.78

RW=4

0.09, 0.23, 0.68

RW=3

0.08, 0.35, 0.57

RW=2

0.1, 0.5, 0.4

RW=1

0.09, 0.74, 0.17

*Figure 1*

Consider the test error after one step split: there are 125 data of test set and each of subset for RW=1, 2,3,4,5, there are 25 data. The genuine number of each class and predicted class are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| RW=1 | | | |
|  | No. of Predicted | No. of Genuine | misclassification |
| B | 2 | 1 | 5 |
| L | 19 | 24 |
| R | 4 | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| RW=2 | | | |
|  | No. of Predicted | No. of Genuine | misclassification |
| B | 2 | 1 | 8 |
| L | 13 | 21 |
| R | 10 | 3 |

|  |  |  |  |
| --- | --- | --- | --- |
| RW=3 | | | |
|  | No. of Predicted | No. of Genuine | misclassification |
| B | 2 | 1 | 9 |
| L | 9 | 18 |
| R | 14 | 6 |

|  |  |  |  |
| --- | --- | --- | --- |
| RW=4 | | | |
|  | No. of Predicted | No. of Genuine | misclassification |
| B | 2 | 1 | 9 |
| L | 6 | 15 |
| R | 17 | 9 |

|  |  |  |  |
| --- | --- | --- | --- |
| RW=5 | | | |
|  | No. of Predicted | No. of Genuine | misclassification |
| B | 1 | 5 | 10 |
| L | 5 | 11 |
| R | 19 | 9 |

The test error after one step=(5+8+9+9+10)/125=32.8%

Analysis: the reason why test error is large is that when I choose attribute 3 as rule to split the tree, the probability distribution of B,L,R in data subset of RW is 8%, 40%, 52%, respectively, while the probability distribution of B,L,R in whole data set is 7%, 46.08%, 46.08%. It means that I have more data belongs to R so that there are a number of data predicted to be R rather than B and L. Thus, major misclassification is that predicted R. It is easy to see from the above tables.

Currently, we have five nodes and then continue to split. Since all records in current data subset do not have the same output, so every node should be split.

For each subsets, continue to calculate which attributes (except attribute 3) have the largest RIG, and continue splitting the tree.

For subset 1 (RW=1)

RIG for attribute 1 (LW)=0.1440; RIG for attribute 2 (LD)=0.2055;

RIG for attribute 4 (RD)=0.1279; choose attribute 2 to split.

For subset 2 (RW=2)

RIG for attribute 1 (LW)=0.1493; RIG for attribute 2 (LD)=0.1717;

RIG for attribute 4 (RD)=0.1356; choose attribute 2 to split.

For subset 3 (RW=3)

RIG for attribute 1 (LW)=0.1503; RIG for attribute 2 (LD)=0.1578;

RIG for attribute 4 (RD)=0.1641; choose attribute 4 to split.

For subset 4 (RW=4)

RIG for attribute 1 (LW)=0.1212; RIG for attribute 2 (LD)=0.1508;

RIG for attribute 4 (RD)=0.2197; choose attribute 4 to split.

For subset 5 (RW=5)

RIG for attribute 1 (LW)=0.1253; RIG for attribute 2 (LD)=0.2187;

RIG for attribute 4 (RD)=0.2688; choose attribute 4 to split.

The second step decision tree is as follows:

Root

0.08(B),0.4(L),0.52(R)

RW=5

0.04, 0.18, 0.78

RW=4

0.09, 0.23, 0.68

RW=3

0.08, 0.35, 0.57

RW=2

0.1, 0.5, 0.4

RW=1

0.09, 0.74, 0.17

LD=5

0.05, 0.95, 0

LD=4

0.05, 0.9, 0.05

LD=3

0.05, 0.85, 0.1

LD=1

0.2, 0.3, 0.5

LD=2

0.1, 0.7, 0.2

*Figure 2*

We can see from figure 2 that the decision tree has many branches at each node. At present, for every RW=i (i=1 to 5), it has five subsets. As mentioned before, using which attribute to split the tree depends on the most informative attributes. For example, RW=1, the attribute of largest RIG is attribute 2, LD.

When we stop? (RIG=0 except attributes that have been analyzed)

In other words, we can judge from probability distribution of the largest RIG:

If the probability distribution has 1 which means we have already got the absolute answer, there is no need to continue classifying. So stop here. Or the subset has only 3 data and are equally distributed (it seems impossible since the number of data in three classes are different), or 2 data and are equally distributed (it can only happen between class L and class R as they have same data in the origin dataset).

Continue to split subset LD=1, the largest RIG is 0.3776, attribute 4: RD.

LD=1

0.2, 0.3, 0.5

RD=5

1, 0 ,0

RD=4

0.25, 0.75, 0

RD=2

0.25, 0.5, 0.25

RD=1

0.25, 0.75, 0

RD=3

0.25, 0.25, 0.5

*Figure 3*

Up to now, we have used three attributes to split the decision tree. For this branch: RW=1---LD=1---RD, only have one attribute (LW) to conduct next step. Like RD=5 as figure 3 shows, we do not need to split anymore. All outputs have the same class. Moreover, for RD=4, there is no data which can be classified into class R. Thus, in the next step splitting this node, we do not need to consider LW=5.

List other four value of LD:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | LD=2 (largest RIG=0.4074) attribute 1: LW | | | | |
| LW=1 | LW=2 | LW=3 | LW=4 | LW=5 |
| Prob. of B | 0.2 | 0.2 | 1 | 1 | No value=5 |
| Prob. of L | 0.2 | 0.6 | 0 | 0 |
| Prob. of R | 0.6 | 0.2 | 0 | 0 |
|  | LD=3 (largest RIG=0.4911) attribute 1: LW | | | | |
| LW=1 | LW=2 | LW=3 | LW=4 | LW=5 |
| Prob. of B | 0.2 | 1 | 1 | 1 | No value=5 |
| Prob. of L | 0.4 | 0 | 0 | 0 |
| Prob. of R | 0.4 | 0 | 0 | 0 |
|  | LD=4 (largest RIG=0.4297) attribute 4: RD | | | | |
| RD=1 | RD=2 | RD=3 | RD=4 | RD=5 |
| Prob. of B | 1 | 1 | 1 | 0.25 | 0.75 |
| Prob. of L | 0 | 0 | 0 | 0.75 | 0.25 |
| Prob. of R | 0 | 0 | 0 | 0 | 0 |
|  | LD=5 (largest RIG=0.4335) attribute 4: RD | | | | |
| RD=1 | RD=2 | RD=3 | RD=4 | RD=5 |
| Prob. of B | 1 | 1 | 1 | 1 | 0.25 |
| Prob. of L | 0 | 0 | 0 | 0 | 0.75 |
| Prob. of R | No data belongs to R | | | | |

As the table above shows,

RW=1---LD=2---LW, and for LW=3, LW=4, all outputs have the same class, so stop here.

RW=1---LD=3---LW, and for LW=2, LW=3, LW=4, all outputs have the same class, so stop here.

RW=1---LD=4---RD, and for RD=1, RD=2, RD=3, all outputs have the same class, so stop here.

RW=1---LD=5---LW, and for RD=1, RD=2, RD=3,LD=4 all outputs have the same class, so stop here.

RD=1

0.25, 0.75, 0

LW=3

1, 0, 0

LW=2

1, 0, 0

LW=4

1, 0, 0

LW=1

1, 0, 0

*Figure 4*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | RD =2 LW | | | | |
| LW=1 | LW=2 | LW=3 | LW=4 | LW=5 |
| Prob. of B | 1 | 1 | 1 | 1 | No value=5 |
| Prob. of L | 0 | 0 | 0 | 0 |
| Prob. of R | 0 | 0 | 0 | 0 |
|  | RD =3 LW | | | | |
| LW =1 | LW =2 | LW =3 | LW =4 | LW=5 |
| Prob. of B | 1 | 1 | 1 | 1 | No value=5 |
| Prob. of L | 0 | 0 | 0 | 0 |
| Prob. of R | 0 | 0 | 0 | 0 |
|  | RD =4 LW | | | |
| LW =1 | LW =2 | LW =3 | LW =4 |
| Prob. of B | 1 | 1 | 1 | 1 |
| Prob. of L | No data belongs to L | | | |
| Prob. of R | 0 | 0 | 0 | 0 |

It is the last step for RW=1---LD=1---RD=1---LW this branch. Clearly, the final outcome is easy to see. Every subset has same output.

Root

0.08(B),0.4(L),0.52(R)

RW=1

0.09, 0.74, 0.17

RW=2

0.1, 0.5, 0.4

RW=3

0.08, 0.35, 0.57

RW=4

0.09, 0.23, 0.68

RW=5

0.04, 0.18, 0.78

LD=2

0.1, 0.7, 0.2

LD=3

0.05, 0.85, 0.1

LD=4

0.05, 0.9, 0.05

LD=5

0.05, 0.95, 0

LD=1

0.2, 0.3, 0.5

RD=2

0.25, 0.5, 0.25

RD=3

0.25, 0.25, 0.5

RD=1

0.25, 0.75, 0

RD=4

0.25, 0.75, 0

RD=5

1, 0 ,0

LW=3

1, 0, 0

LW=2

1, 0, 0

LW=4

1, 0, 0

LW=1

1, 0, 0

*Figure 5*

We can build the whole decision tree following these steps. But in every time splitting a node, we need “for” loops. Since there are 4 attributes and each attribute has 5 values; If not writing loops to help us program, the task is trivial and time-consuming. However, the problem is that how to store many m\*n matrixes. Such as, at root, we have 500 data, 500\*5, as learning set. Then, for each RW=i (i=1 to 5) subset, there are 100 data, 100\*5 in each subset (except attribute 3 we have used, 100\*5 can be condensed into 100\*4, three unused attribute and one class label, but 100\*5 is ok to continue next step only RIG(attribute3)=0). Next, we need a loop to calculate the largest RIG for RW=i (i=1 to 5) and many other related data, but I am able to implement this as I have no idea how to store 5 subsets which consists of 100\*5 of each. Because I cannot handle this, I only have this path RW=1---LD=1---RD=1---LW=1,2,3,4,5 and I can only calculate the first-step test error.

1. Summarization : Three aspects of cognitive development

The goal of this article is to make both conceptual and experimental distinctions among the three domains (specific knowledge governing task performance, responsiveness to experience, and basic processes that underlie differences in the other two areas) and to map out the interrelationships among them.

In order to analyze specifically, they devise three experiments.

Experiment 1: models of children’s specific knowledge about balance scale problems.

The goal of this experiment is to determine whether children’s knowledge about balance scale problems could be characterized accurately and unambiguously. The author states four rules and through this experiment found that the accuracy of rules. There are 107 children can be classified as one of the four rules. Moreover, experiment 1 illustrates the advantage of the four rules. The advantage over the more traditional general stage approach was in the precision with which the developmental progression was portrayed and in the number and concreteness of the predictions. Another finding of experiment 1 is that children are not likely to use rule 4 since lack of rule 4 in their experience.

Experiment 2: responsiveness to experience

The purpose of experiment 2 is to verify that children using rule 1 would benefit more from experience with distance problems than from experience with conflict problems. In specific, whether age-related factors should be considered while assessing children’s knowledge about a problem. In conclusion, they found that compared to older children, younger children (five years old) are less able to acquire new information since their ability to grasp information is less adequate.

Experiment 3a: the encoding hypothesis

The purpose of experiment 3a is to verify that older children are likely to encode on both weight and distance dimensions. It is relatively accurate. While younger children presumed to encode on weight but not on distance.

Experiment 3b:

To give younger children more time, they can encode the distance dimension or not.

Experiment 3c:

To verify that another possible reason leading to the differences in patterns of encoding is because younger children cannot understand the instruction wholly.

In sum, younger children and older children react differently to encode balance problem. However, misunderstanding of the instructions cannot affect the result of experiment so it cannot be taken into account.

1. Summarization: Modeling cognitive development on balance scale phenomena

f

2 Car evaluation

The difference between this task and last task is only the value of input attributes. Car evaluation has 6 input attributes and 4 of them are nominal value which is different from last task’s numeric value. There are 6 attributes evaluating the performance of car:

1. Buying buying price
2. Maint price of the maintenance
3. Doors number of doors
4. Persons capacity in terms of persons to carry
5. Lug\_boot the size of luggage boot
6. Safety estimated safety of the car

Attribute values:

Buying: v-high, high, med, low (4 values)

Maint: v-high, high, med, low (4 values)

Doors: 2, 3, 4, 5-more (4 values)

Persons: 2, 4, more (3 values)

Lug\_boot: small, med, big (3 values)

Safety: low, med, high (3 values)

Class distribution:

Unacc 1210 (70.023 %)

Acc 384 (22.222 %)

Good 69 ( 3.993 %)

V-good 65 ( 3.762 %)

Because lowercase is not convenient for Matlab to read. One of the methods is what I have used above. For every attribute, choosing different number represents the value in this attribute. For instance, attribute 1= Buying: v-high=1, high=2, med=3, low=4. This might be a feasible solution.

However, this task data has more attributes to deal with so this seems to be more challenging than the former one. As I mentioned before about the drawback of my method, it is not possible to draw the whole tree. When I master the way to store multi-dimension data, I may complete the task. But now, I can only draw the first-step decision tree and try to find test-error. From lecture notes, the training error can be seen only the whole tree is finished. Therefore, I did not find the training error of balance scale classification.

Here, we use the location of first lowercase in 24 letters to represent label. Class unacc=21, acc=1, good=7, v-good=22; Buying attribute value: v-high=22, high=8,

med=13, low=12; Doors: 2, 3, 4, 5-more=5; Persons: 2, 4, more=5; Lug\_boot: small=19, med=13, big =2; Safety: low=12, med=13, high=8 .

Next, choosing the attribute which has the largest relative information gain is a node to split. According to the matlab result:

Attribute 1: Buying

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| vaule | high | low | med | v-high | RIG |
| acc | 0.25 | 0.3023 | 0.2662 | 0.1667 | 0.0718 |
| good | 0.75 | 0.6977 | 0.0532 | 0.8333 |
| Unacc | 0 | 0 | 0.6204 | 0 |
| v-good | 0 | 0 | 0.0602 | 0 |

Attribute 2: Maint

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| vaule | high | low | med | v-high | RIG |
| acc | 0.2222 | 0.2531 | 0.3241 | 0.1512 | 0.0750 |
| good | 0.7778 | 0.0710 | 0.6358 | 0.8488 |
| Unacc | 0 | 0.6358 | 0.0401 | 0 |
| v-good | 0 | 0.0401 | 0 | 0 |

Attribute 3: Doors

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| vaule | 2 | 3 | 4 | 5-more | RIG |
| acc | 0.1909 | 0.2393 | 0.2564 | 0.2432 | 0.004 |
| good | 0.0142 | 0.0171 | 0.0171 | 0.0182 |
| Unacc | 0.7835 | 0.7265 | 0.7037 | 0.7143 |
| v-good | 0.0114 | 0.0171 | 0.0228 | 0.0243 |

Attribute 4: persons

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| vaule | 2 | 4 | 5-more | RIG |
| acc | 1 | 0.3551 | 0.3442 | 0.1876 |
| good | 0 | 0.0261 | 0.024 |
| Unacc | 0 | 0.5926 | 0.6013 |
| v-good | 0 | 0.0261 | 0.0305 |

Attribute 5: Lug\_boot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| vaule | big | med | small | RIG |
| acc | 0.2876 | 0.2473 | 0.1623 | 0.0245 |
| good | 0.0174 | 0.0174 | 0.0152 |
| Unacc | 0.6601 | 0.7137 | 0.8225 |
| v-good | 0.0349 | 0.0217 | 0 |

Attribute 6: Safety

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| vaule | high | low | med | RIG |
| acc | 0.4087 | 1 | 0.2885 | 0.2151 |
| good | 0.0217 | 0 | 0.0282 |
| Unacc | 0.5130 | 0 | 0.6833 |
| v-good | 0.0565 | 0 | 0 |

Clearly, for the first-step, attribute 6 has the largest RIG.

Root

0.23, 0.0166, 0.7323, 0.0188

Safety=med

0.2885, 0.0282, 0.6833,0

Safety=high

0.4087, 0.0217, 0.513,0.0565

Safety=low

1,0,0,0

Moreover, safety=low this node do not need to be split any more. We can use the method from the former task to calculate the test error for the first-step decision tree.

**Reference**

Lecture notes

Data Mining and Knowledge Discovery Handbook; Edited by Oded Maimon, Lior Rokach, Springer, 2005

psychological experiments with human learning [1] and machine learning [2] for this classification problem.[1] Siegler, R. S. (1976).

**Appendix**

1. Balance scale classification

close all;

clear;

clc;

format compact;

load decision\_tree.mat;

o\_data=a;

[m,n]=size(o\_data);t=round(m\*0.8);l\_data=o\_data(randperm(t),:);t\_data=o\_data(~ismember(o\_data,l\_data,'rows'),:);

[a,b,z]=jud\_(l\_data,4)

a1=l\_data(find(l\_data(:,b)==1),:);t1=t\_data(find(t\_data(:,b)==1),:);

a2=l\_data(find(l\_data(:,b)==2),:);t2=t\_data(find(t\_data(:,b)==2),:);

a3=l\_data(find(l\_data(:,b)==3),:);t3=t\_data(find(t\_data(:,b)==3),:);

a4=l\_data(find(l\_data(:,b)==4),:);t4=t\_data(find(t\_data(:,b)==4),:);

a5=l\_data(find(l\_data(:,b)==5),:);t5=t\_data(find(t\_data(:,b)==5),:);

[a11,b11,z11]=jud\_(a1,4);

[a12,b12,z12]=jud\_(a2,4);

[a13,b13,z13]=jud\_(a3,4);

[a14,b14,z14]=jud\_(a4,4);

[a15,b15,z15]=jud\_(a5,4);

l1=a1(find(a1(:,b11)==1),:);

l2=a1(find(a1(:,b11)==2),:);

l3=a1(find(a1(:,b11)==3),:);

l4=a1(find(a1(:,b11)==4),:);

l5=a1(find(a1(:,b11)==5),:);

[l11,k11,r11]=jud\_(l1,4)

[l12,k12,r12]=jud\_(l2,4)

[l13,k13,r13]=jud\_(l3,4)

[l14,k14,r14]=jud\_(l4,4)

[l15,k15,r15]=jud\_(l5,4)

x1=l1(find(l1(:,k11)==1),:);

x2=l1(find(l1(:,k11)==2),:);

x3=l1(find(l1(:,k11)==3),:);

x4=l1(find(l1(:,k11)==4),:);

x5=l1(find(l1(:,k11)==5),:);

[m11,n11,w11]=jud\_(x1,4)

[m12,n12,w12]=jud\_(x2,4)

[m13,n13,w13]=jud\_(x3,4)

[m14,n14,w14]=jud\_(x4,4)

[m15,n15,w15]=jud\_(x5,4)

%find the largest RIG

function [a,b,z\_index]=jud\_(data,index)

for i=1:index

% RIG(i)=rig(data(:,i),data(:,5));

RIG(i)=rig(data(:,i),data(:,7));

end

[a,b]=max(RIG);

z\_index=find(RIG==0);

%calculate RIG

function RIG=rig(x,y)

b=unique(y);

c=zeros(size(b));

for i=1:length(b)

c(i)=length(find(y==b(i)));

end

p=c./sum(c)

Hy=0;

for i=1:length(b)

Hy=Hy+(-p(i)\*log2(p(i)));

end

Hy;

xy=[x,y];

b1=unique(xy(:,1))

c1=zeros(size(b1));

for i=1:length(b1)

c1(i)=length(find(xy(:,1)==b1(i)));

end

p1=c1./sum(c1)

for i=1:length(y)

for j=1:length(b1)

if xy(i,1)==b1(j)

temp(i,j)=xy(i,2);

end

end

end

p11=zeros(length(b),length(b1));

for i=1:length(b1) %b1=5

temp1=temp(temp(:,i)~=0,i);

b11=unique(temp1)

c11=zeros(size(b11));

for j=1:length(b11)

c11(j)=length(find(temp1==b11(j)));

end

p11([1:length(b11)],i)=c11./sum(c11);

end

p11

[m,n]=size(p11);

Hyx=zeros(1,n);

for i=1:n

for j=1:m

if p11(j,i)==0

p11(j,i)=1;

end

Hyx(i)=Hyx(i)+(-p11(j,i)\*log2(p11(j,i)));

end

end

HYX=Hyx\*p1;

IG=(Hy-HYX);

RIG=IG/Hy

2. car evaluation

close all;

clear;

clc;

format compact;

load car.mat;

o\_data=b;

[m,n]=size(o\_data);t=round(m\*0.8);l\_data=o\_data(randperm(t),:);t\_data=o\_data(~ismember(o\_data,l\_data,'rows'),:);

[a,b,z]=jud\_(l\_data,6)